**implementasi highchart dalam yii framework**

24 October 2013[mrkoeh](https://kukuhnugroho.wordpress.com/author/mrkoeh/)[Leave a comment](https://kukuhnugroho.wordpress.com/2013/10/24/implementasi-highchart-dalam-yii-framework/#respond)[Go to comments](https://kukuhnugroho.wordpress.com/2013/10/24/implementasi-highchart-dalam-yii-framework/#comments)

Pada kesempatan kali ini saya akan memberikan sedikit tips tentang bagaimana mengimplementasikan highchart dalam yii framework.

Kadang, dalam sebuah aplikasi kita diharuskan untuk menampilkan sebuah laporan dengan menggunakan grafik/chart.

Kali ini kita akan mencoba menggunakan [highchart](http://www.highcharts.com/).

Pertama kita download terlebih dahulu [extension highchart](http://www.yiiframework.com/extension/highcharts/) dan masukkan dalam folder**protected/extension.**

Setelah itu kita buat script di dalam controller

|  |  |
| --- | --- |
| 123456789101112 | public function actionGrafik() { $sql='SELECT \* FROM pendapatanrs ORDER BY id\_pendapatan ASC';  $dataProvider=new CSqlDataProvider($sql,array( 'keyField' => 'id\_pendapatan', )); $this->render('grafik',array( 'dataProvider'=>$dataProvider, )); } |

script di view

|  |  |
| --- | --- |
| 123456789101112131415161718192021222324252627282930313233343536373839404142434445464748495051525354 | <?php $form=$this->beginWidget('CActiveForm', array( 'id'=>'tinstrument-form', 'enableAjaxValidation'=>false,)); ?> <center> <?php$label=array();$nilai=array(); foreach($dataProvider->getData() as $i=>$ii){ $label[$i]=$ii['keterangan']; $nilai[$i]=doubleval($ii['jumlah']);} $this->widget('application.extensions.highcharts.HighchartsWidget', array( 'options'=>array( 'chart'=> array('defaultSeriesType'=>'column',), 'title' => array('text' => 'Pendapatan Rumah Sakit'), 'legend'=>array('enabled'=>false), 'xAxis'=>array('categories'=>$label, 'title'=>array('text'=>'Keterangan'),), 'yAxis'=> array( 'min'=> 0, 'title'=> array( 'text'=>'Jumlah' ), ), 'series' => array( array('name' => $label,'data' => $nilai) ), 'tooltip' => array('formatter' => 'js:function(){ return "<b>"+this.point.name+"</b> :"+this.y; }'), 'tooltip' => array( 'formatter' => 'js:function() {return "<b>"+ this.x +"</b><br/>"+"Jumlah : "+ this.y; }' ), 'plotOptions'=>array('pie'=>(array( 'allowPointSelect'=>true, 'showInLegend'=>true, 'cursor'=>'pointer', ) ) ), 'credits'=>array('enabled'=>false), ))); ?></center> <?php $this->endWidget(); ?> |

dan terjadilaah. . .hehehe

untuk menampilkan pie chart

|  |  |
| --- | --- |
| 12345678910111213141516171819202122232425262728293031323334353637383940 | <?php $form=$this->beginWidget('CActiveForm', array('id'=>'tinstrument-form','enableAjaxValidation'=>false,)); ?> <center><?php $bb=array();foreach($dataProvider->getData() as $i=>$ii) $bb[$i]=array($ii['keterangan'],doubleval($ii['jumlah'])); $this->widget('application.extensions.highcharts.HighchartsWidget', array( 'options'=>array( 'series' => array( array('type'=>'pie', 'data' => $bb ) ), 'title' => array('text' => 'Pendapatan Rumah Sakit'), 'tooltip' => array( 'formatter' => 'js:function(){ return "<b>"+this.point.name+"</b> :"+this.y; }' ), 'plotOptions'=>array('pie'=>(array( 'allowPointSelect'=>true, 'showInLegend'=>true, 'cursor'=>'pointer', ) ) ), 'credits'=>array('enabled'=>false), ))); ?></center> <?php $this->endWidget(); ?> |

# **Handling Statistics – Pulling data from Yii and displaying with HighCharts**

posted in [Yii Tutorials](http://www.ponomaryov.org/category/yii-tutorials/) on [August 17, 2014](http://www.ponomaryov.org/yii-tutorials/handling-statistics-pulling-data-from-yii-and-displaying-with-highcharts/) by [Serge Ponomaryov](http://www.ponomaryov.org/author/admin/)

 SHARE

“Facts are stubborn things, but statistics are pliable.”
― Mark Twain

Sometimes, you might want to display some kind of charts in your Yii project based on the data that you’ve collected in your database. From the front end, there is an awesome way to do so, being a plugin called HighCharts, that is both eye-appealing and free to use if you’re not a commercial website. In this post, we’ll be looking at how to integrate it with a back end running on Yii. For an example of data, we’ll be logging website visits and visitors.

**Gathering data**

First of all, let’s get some data to start with. Create a simple table in the database that looks like this:



and generate a model for it using Gii.

Now, we’ll be adding a mechanism to record the visits. It should be sitewide, so let’s add it to components/Controller.php. It should also determine whether the user is a new visitor or someone who we already have in our records. We’ll be using a permanent cookie for that.

The code is pretty straightforward:

public function init() {

if(isset(Yii::app()->request->cookies['user\_identifier']->value)) {

// returning visitor

$user\_identifier = Yii::app()->request->cookies['user\_identifier']->value;

// try to find if we have anything logged for today

$model = Visitors::model()->findByAttributes(array("date"=>date("Y-m-d"), "user\_identifier"=>$user\_identifier));

if(!$model) {

// first time today!

$model = new Visitors;

$model->date = date("Y-m-d");

$model->user\_identifier = $user\_identifier;

$model->visits = 1;

$model->save();

}

else {

// just adjust the count

$model->visits++;

$model->save();

}

}

else {

// a new visitor

$user\_identifier = md5(rand()); // a nice way to generate a random string

$cookie = new CHttpCookie('user\_identifier', $user\_identifier);

$cookie->expire = time() + (60\*60\*24\*365\*5); // should not expire less than in five years!

Yii::app()->request->cookies['user\_identifier'] = $cookie;

// record the visit

$model = new Visitors;

$model->date = date("Y-m-d");

$model->user\_identifier = $user\_identifier;

$model->visits = 1;

$model->save();

}

parent::init();

}

Now, if you get back to the site and open up a few pages, you should get a single record in the database. In my case, it looks like this:



**Setting up HighCharts**

So far, so good. Now, let’s render some actual charts! You can grab the plugin in question at [http://highcharts.com](http://highcharts.com/) and copy all the files to your project, or simply add a few lines to the main layout:

<script src="http://code.jquery.com/jquery-1.11.1.js"></script>

<script src="http://code.highcharts.com/highcharts.js"></script>

<script src="http://code.highcharts.com/modules/exporting.js"></script>

Once that is done, we should tell it to render something! There’s dozens of demos for different kinds of graphs you can have with HighCharts on[this page](http://www.highcharts.com/demo), but we will be using a plain line chart for now.

Take the code from this [JSFiddle](http://jsfiddle.net/gh/get/jquery/1.9.1/highslide-software/highcharts.com/tree/master/samples/highcharts/demo/line-basic/) and add it to a view you want to use. I will be using the site index. Don’t forget to add a container div that you will be rendering in!

And there it goes:



Okay, but how do you get the data to be dynamic? Changing it right in the view, according to MVC, is not the proper approach to doing so. Why don’t we try grabbing it with JSON then? Wrap the highcharts init call in agetJSON function:

$.getJSON('http://www.highcharts.com/samples/data/jsonp.php?filename=analytics.csv&callback=?', function (csv) {

Then our code will look like this (let’s also change some labels, while we’re at it!)

<script>

$(function () {

$.getJSON('<?php echo $this->createUrl("site/stats"); ?>', function (json) {

$('#container').highcharts({

title: {

text: 'Site Traffic',

x: -20 //center

},

xAxis: {

categories: json.dates

},

yAxis: {

title: {

text: 'Daily count'

},

plotLines: [{

value: 0,

width: 1,

color: '#808080'

}]

},

legend: {

layout: 'vertical',

align: 'right',

verticalAlign: 'middle',

borderWidth: 0

},

series: [{

name: 'Unique visitors',

data: json.visitors

}, {

name: 'Page visits',

data: json.visits

}]

});

});

});

</script>

<div id="container"></div>

This will be initiating an AJAX call to actionStats in our SiteController. Such an action is not yet there, though. Let’s go ahead and create one.

public function actionStats()

{

$response = array();

// collect all the unique dates

$dates = array();

$models=Visitors::model()->findAll(array(

'select'=>'date',

'group'=>'date',

'distinct'=>true,

));

foreach($models as $model)

array\_push($dates, $model->date);

// why not also format the dates for display?

$response["dates"] = array();

foreach($dates as $date) {

$datetime = new DateTime($date);

array\_push($response["dates"], $datetime->format("M jS"));

}

// count the visitors and the visits for each day

$response["visitors"] = array();

$response["visits"] = array();

foreach($dates as $date) {

$visits = 0;

$visitors = Visitors::model()->findAllByAttributes(array("date"=>$date));

foreach($visitors as $visitor)

$visits += $visitor->visits;

array\_push($response["visitors"], count($visitors));

array\_push($response["visits"], $visits);

}

echo json\_encode($response);

}

And that will give us a JSON response just like this:

{"dates":["2014-08-17"],"visitors":[2],"visits":[17]}

Why don’t we cheat with the statistics now and add a few other dates manually through the database, so we have several?

Then, if we refresh the website, we’ll have our nice-looking graphs right there.



The picture is also easily exported to several formats, including PDF and SVG.

Basing on this, you can create almost any kinds of charts that you need for your website. Just be sure to keep your code clean and follow the MVC practices!